GDA at Diamond Light Source 2010

Paul Gibbons PhD., Data Acquisition Team Leader, Diamond Light Source, UK

Presentation Outline

- Diamond - a brief description
- The Diamond Data Acquisition System
- Why use GDA -the demands placed on the data acquisition system
- What is GDA - what does it provide
- How does GDA work
- GDA's weaknesses
- Users response to GDA at Diamond
- Experience of developing and supporting GDA at Diamond
- What next?
- Slide show and Demo

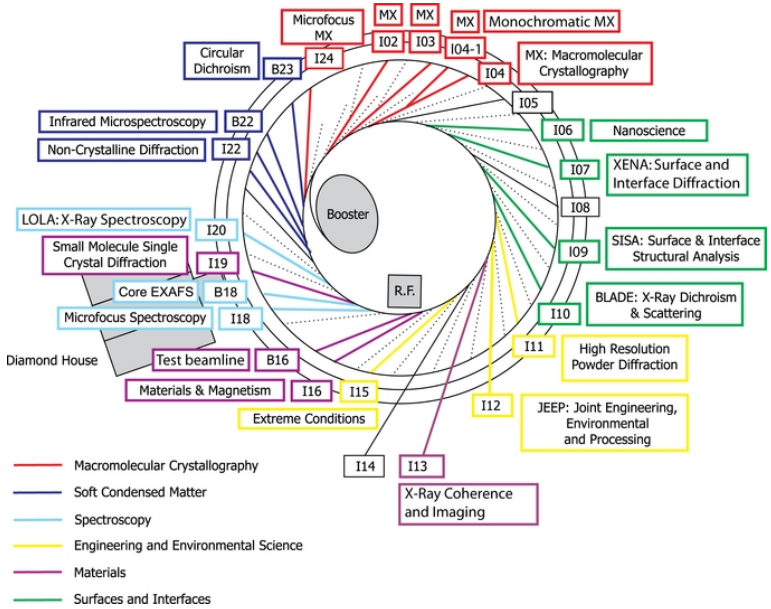
Diamond - outside

Diamond - inside

Diamond - key facts

- 24 cell 3 Gev machine
- 300 mA in top up mode
- 14 beamlines operational
- 22 beamlines by 2012
- A further 10 by 2016

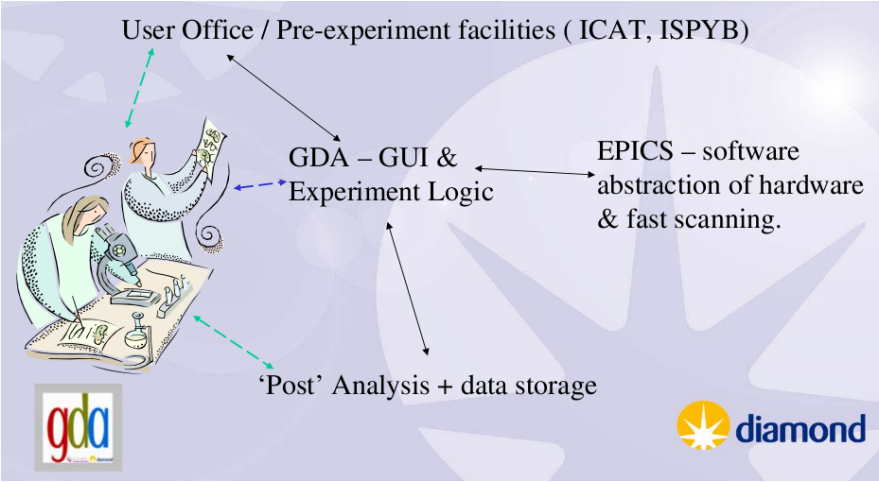
Diamond - beamline layout



For more details go to http://www.diamond.ac.uk/Home/Beamlines.html

Diamond Data Acquisition System - Overview

- EPICS provides most hardware interface and low level experiment synchronisation
- GDA integrates the EPICS system and other devices to give the user level functionality



The Data Acquisition has a wide range of use

- large range of science techniques - surface science, MX, tomography, spectroscopy,
- large range of equipment throughout the facility
- high throughput industrial type measurements by users with little need or desire to understand how the measurements are being taken
- longer experiments by users that that want to control every aspect and create novel experiments during their visit.
- large data volumes,
- 24x7 data collection with little support from engineering staff
- Remote Access
- provide tools for commissioning, beam line setup and beamline use

The Data Acquisition must have many different features

- ease of use
- automatic data reduction and visualisation
- tools to aid analysis to allow users to get confidence in quality of data or to direct the current experiment
- use of a common data format to allow automatic interpretation by visualisation and analysis programs
- connection to user databases used to direct and record the measurement/experiment

The Data Acquisition must be adaptable

- flexibility to allow rapid adaptation of techniques by beamline staff rather than by engineers
- need to change equipment rapidly

- a common tool set, structure and user interface to allow users, beamline staff and engineers to easily move from one beamline to another,
- expectations of GUI's are always going up and so GUI's must be easy to change without impacting the beamline operation.

The Data Acquisition must work with other systems

- connection to archival systems
- connection to user databases for authorisation
- logging facilities
- error reporting systems

The Data Acquisition must be cost effective

- The skills required to maintain and enhance it must be readily available
- The technology must be relevant for a very long time
- The system must be maintainability and remain able to keep up with future demands

What is GDA

- One or more ObjectServers each holding collection of Java objects. The objects may themselves provided a connection to external hardware controllers e.g. EPICS devices
- A mechanism for one object to access another across ObjectServers
- A mechanism for objects to notify other objects of changes of state
- One ObjectServer contains a Jython Interpreter This has access to the other objects in the system
- A highly flexible Scan mechanism that can be used to a wide range of of measurements
- Rich GUI clients that also have access to the other objects in the system, including the Jython Interpreter The Rich GUI is constructed using Eclipse plugin framework It also has an ObjectServer embedded in it
- Eclipse Plugins exist to give access to the objects, jython interpreter and provide data visualisation
- For more detail see the Developer and User documentation

How does GDA satisfy the demands placed upon it -(1)

- highly flexible The objects are constructed using Spring Framework config files which places little restrictions on their form or function
- integration - Jython language used to integrate the objects is very powerful
- little engineering support - the system is stable and well tested
- very powerful common scan mechanism that can easily be used to scan any combination of real or virtual devices at any level of nesting

- scan mechanism creates plottable data and common format data files that can cope with any level of complexity of data. But the pipeline for handling scan results can be extended and adapted easily
- scanned devices can be written in Java or Jython, by engineers and beamline users. They can be as simple as returning the beam current or as complex as configuring and reading a multi element Germanium detector , or driving a combined motor trajectory scan with synchronised data measurement

  A scan of energy keeping hkl fixed at [1 0 1] , with polarisation set to 90, at each energy reading the value from a counter timer exposed for 1 second is read, plotted and written to file:

```
scan energy 5.95 6.05 .0005 hkl [1 0 1] pol 90 ct 1
```

How does GDA satisfy the demands placed upon it (2)

- easy to use command line access to a Jython interpreter to allow low level control of the system,
- ability to execute user written Jython scripts to automate lengthy or complex procedures.
- CAJ/JCA built in and GDA helper classes allow easy creation of devices to interface to EPICS pvs
- DiffCalc provides scanning in hkl space by diffractometer in various modes. Diffcalc objects fit perfectly into the scan mechanism

How does GDA satisfy the demands placed upon it (3)

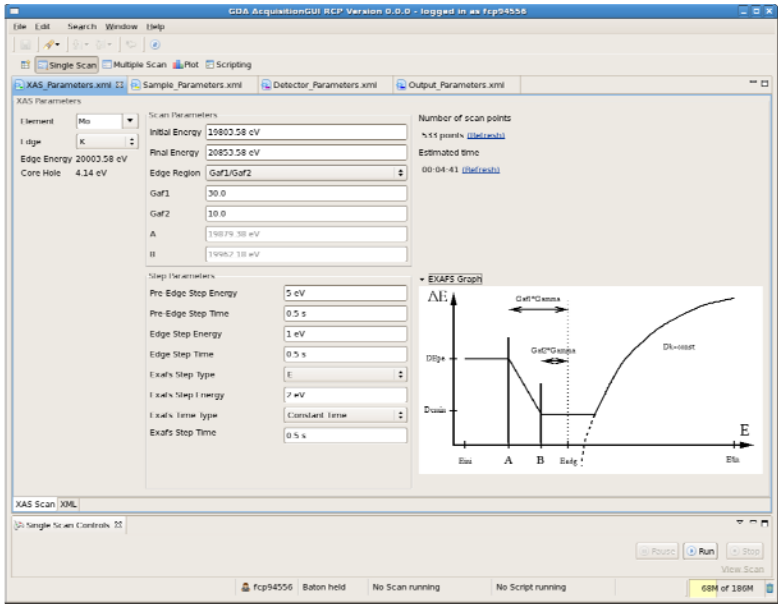
- Client server design to allow multiple users to access the system at once without endangering the data collection process.
- A baton system to control who has ability to change the beamline operation.
- An authorisation system to allow different user levels of control.

How does GDA satisfy the demands placed upon it (4)

- The Eclipse Rich Client Platform plugin framework to allow software engineers to create either technique specific client guis and associated server components, data visualisation components or data reduction pipelines.
- A basic set of GUI panels that allow users to write scripts, enter commands, scan devices and visual results.
- A proven design pattern for interfacing GUI aspects of experiment definition to server script. A growing library of compatible GUI widgets to speed development of the GUI aspects.
- The resultant GUI can be branded to suit wishes on individual beamlines Beamline specific perspectives, manual, splash screen

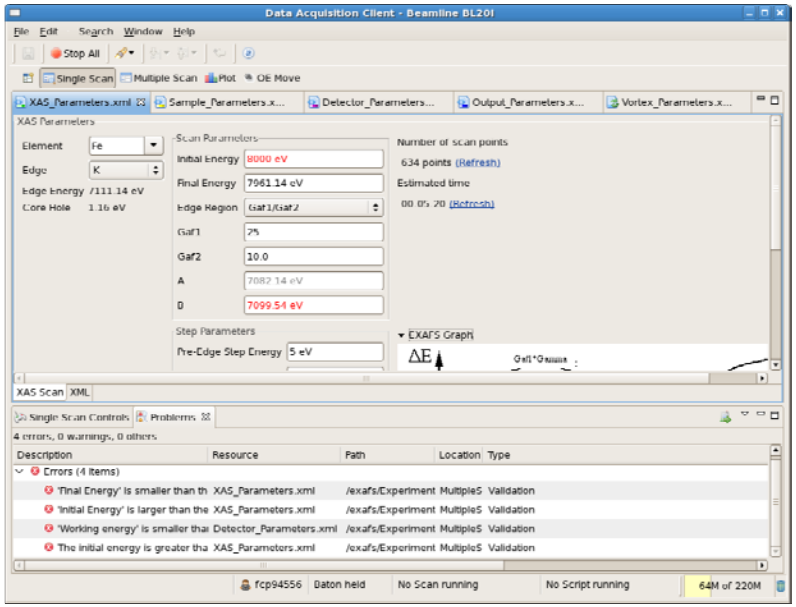
Experiment Definition GUI design - common widgets

The GDA project contains a wide range of higher level widgets based on SWT and JFace. These are included in the common.rcp plugin and are available in GDA 8.2. They are used through a tool palette in RCP developer so can be placed graphically into your UI. More information is available on the GDA forum



Experiment Definition GUI design - built in validation

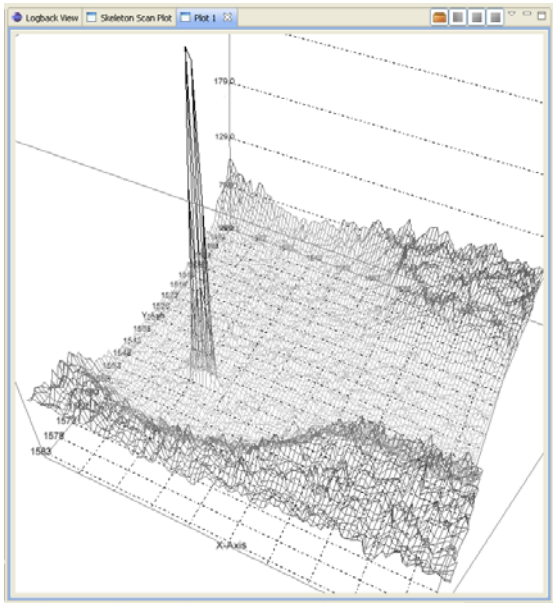
The experiment parameter definition system allows for automatic validation against a XSD schema and Castor XML mapping file as well as editor specific code contributed validation. Errors are reported to the user using Eclipse problem reporting mechanism.



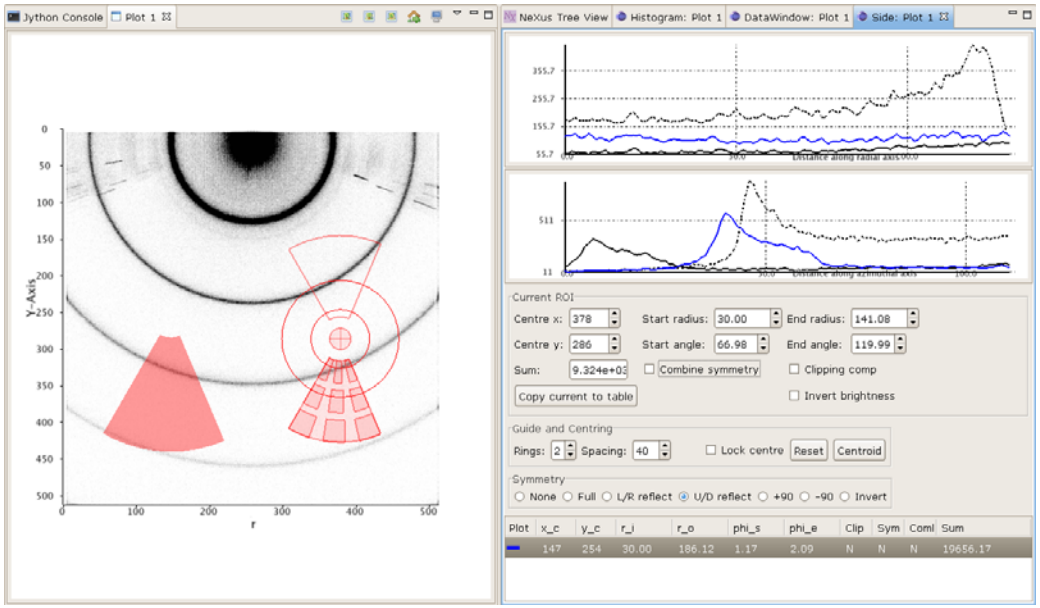
How does GDA satisfy the demands placed upon it (5)

- A growing set of data visualisation tools using OpenGL to make use of hardware capabilities of the system graphics card.
- A set of FileIO tools to read data files of various formats into a common data format (DataSet) for access by the rest of the analysis tools
- The DataSet class aims to emulate some of the functionality of NumPy's ndarray class.

How does GDA satisfy the demands placed upon it - analysis tools



How does GDA satisfy the demands placed upon it - analysis tools

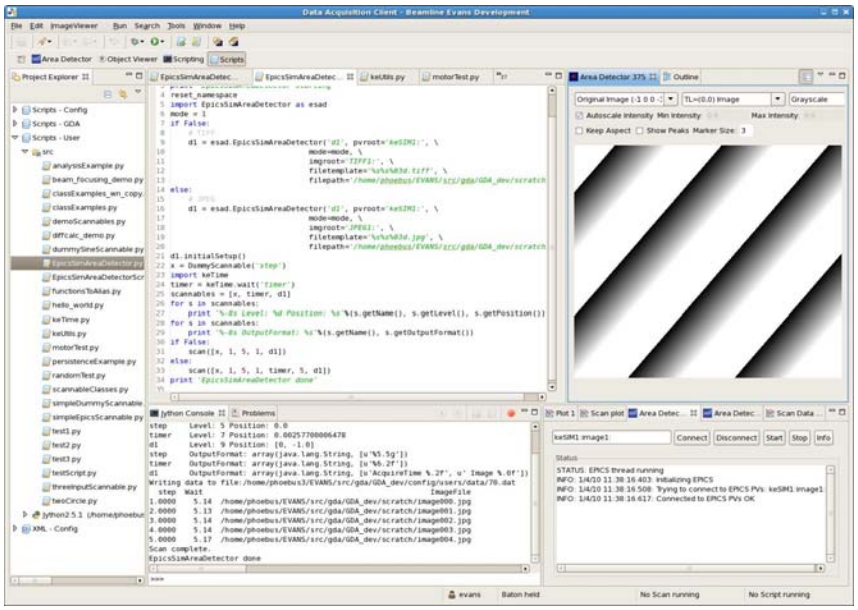


How does GDA satisfy the demands placed upon it (7)

- Being Eclipse based the GUI can easily be augmented by other plugins that are not aware of GDA. Some of these are already built in such as:
  - PyDev python editor;
  - CSS Synoptic Display Studio which allows non software experts to create bespoke panels to talk to both GDA native objects or directly to EPICS.
  - EDNA - analysis pipeline

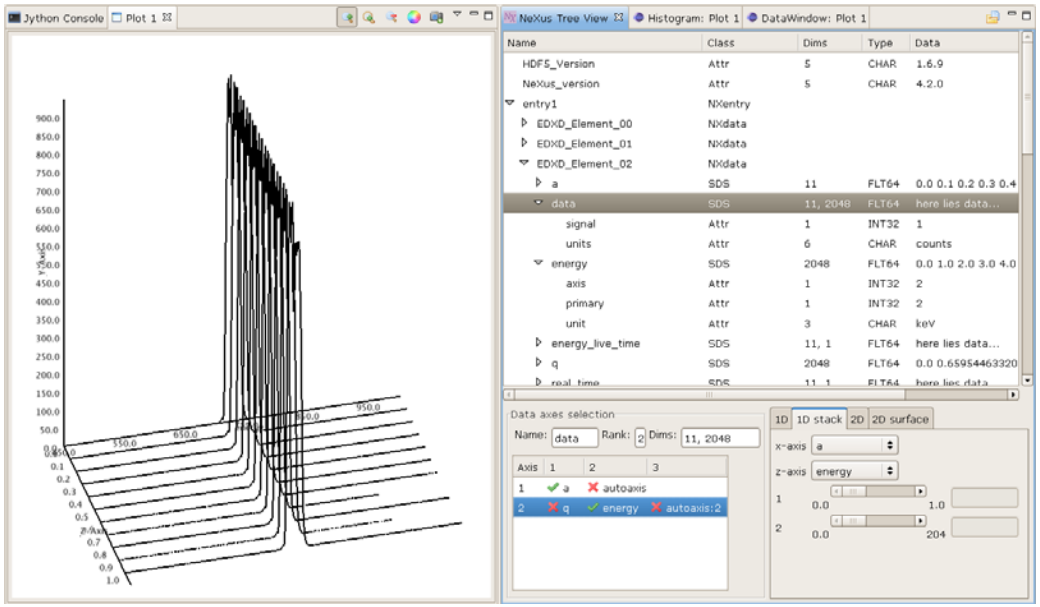
Combining GDA with other plugins

Kenneth Evans at APS has incorporated Fable plugins getting its images from the simulator for Mark Rivers' Area detector into GDA: The following image also shown a Jython script that accesses this detector and outputs five images via the scan command.



How does GDA satisfy the demands placed upon it (6)

- Full support for Nexus file creation and visualisation.



How does GDA satisfy the demands placed upon it (7)

- Use of the log4j logging system to provide system wide logging and program events.

How does GDA satisfy the demands placed upon it (8)

- Being Java based it is easier to find software engineers with required skill set.
- There is a wealth of good Java development tools.
- Use of widely used Java technologies , e.g. Spring Framework, Jython, Eclipse
- Jython is easy for beamline staff and 'expert' users to learn.

Weaknesses

- Not a control system.
- No archiving
- No alarm system
- No strip tool (yet) other than that in CSS-SDS

Users response to using GDA at Diamond

- Users of phase 1 beamlines that continue to use the Swing GUI express a lot of happiness with the system.
- Beamline staff are very happy with the flexibility of the scan system
- The Eclipse GUI is new and is only now being given to users and beamline staff to use in anger. They can appreciate the power of the system but are not yet in a position to comment further. There is a training exercise that needs to take place.

Experience of developing and supporting GDA at Diamond

- Developing script base experiments and incorporation of new device is very simple.
- The basic system just works and can be relied upon. This is the result of improvements to development practises made over the last 2 years.
- Support of the system demands little effort from the engineering team. Most unplanned time on the beamline is spent adapting the system to meet new requirements.
- GUI development has been tedious and time consuming. It is hoped that with the introduction of the Eclipse system, a proven design pattern for linking GUI to script engine and the development of a set common widgets that this task becomes easier.
- However developing sets of Eclipse plugins to deliver an overall system is not easy and requires teamwork, good Java developers, and the following of software engineering practises.

Development Process

- Continuous integration using Hudson
- Unit and integration testing of Java and scripts after each commit
- Well controlled release procedure
- Use of Jira issue tracker

Lessons learnt

- Stability is all important. If it worked once it should continue to work This demands the team adhere to software engineering practises
- Communication with the EPICS developers is very important. Use a mechanism to abstract away pv names behind interface definitions
- Push control down to as close to hardware as possible EPICS trajetory scans that make use of the abilities of the motor controllers are used to provide fast synchronised measurements when needed

The Diamond Data Acquisition and Scientific Computing Group

Bill Pulford - Head of Group

Data Acquisition Team	Diamond Scientific Software
Paul Gibbons (Team Manager)	Alun Ashton (Team Manager)
Jun Aishima	Mark Basham

Vasanthi Nagalingam
Eric Ren
Tobias Richter
Rob Walton
Richard Woolliscroft
Fajin Yuan
Matt Gerring
Matthew Webber
Richard Fearn
Chris Coles
Karl Levik
Peter Chang
Duncan Sneddon
Joachim Diepstraten
Graeme Winter

GDA originated from the team at Daresbury Laboratory

Greg Diakun - Head of Group

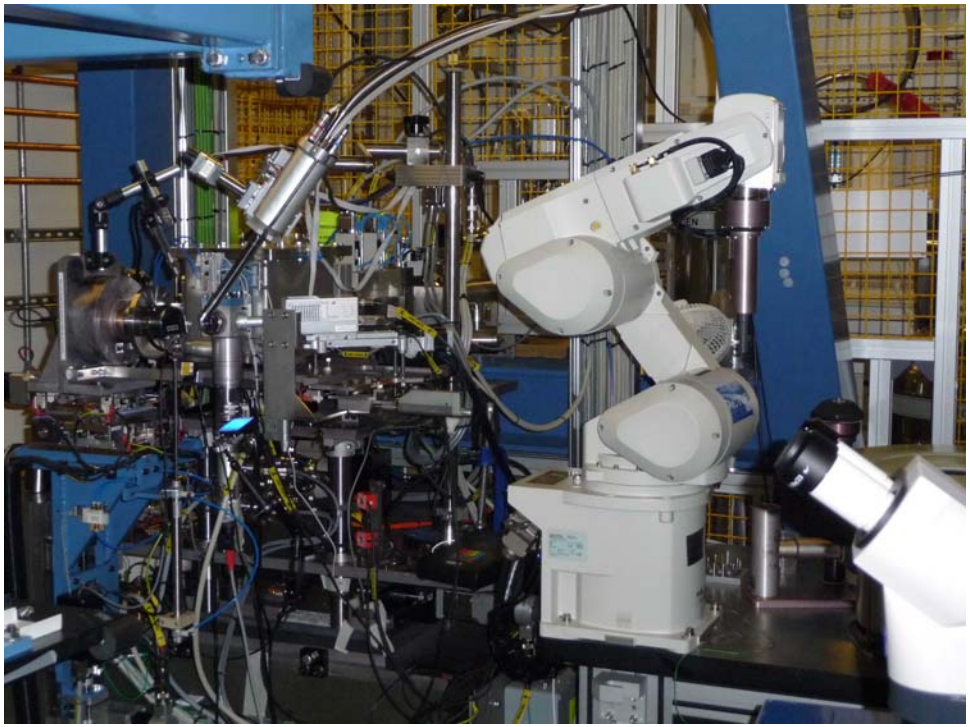
Geoff Mant
Paul Stephenson
Karen Ackroyd
Glenys McBain
Steve Kinder
Christine Ramsdale
Mike Miller

What next for GDA

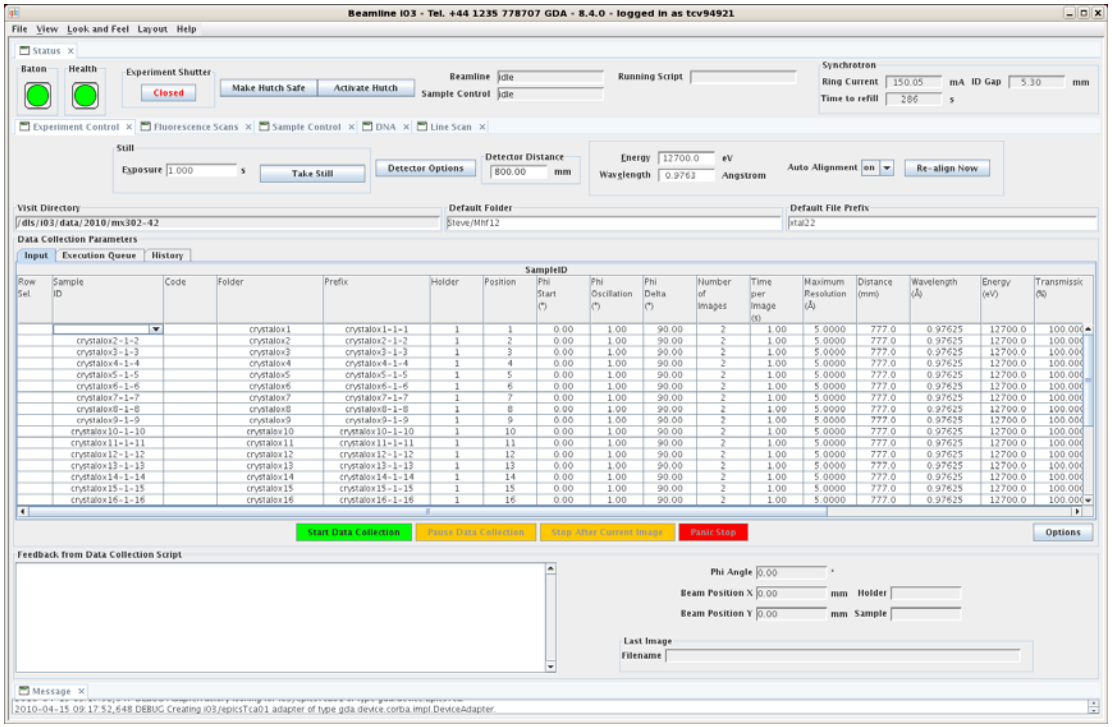
- Converting the MX Gui from Swing to Eclipse RCP
- Development of GUIs, scripts and server objects to support the new beamlines at Diamond
- Incorporation of analysis pipelines
- Make better use of the CSS-SDS.
- Make source code repository open to collaborators Currently we are in favour of using git

MX User Environment

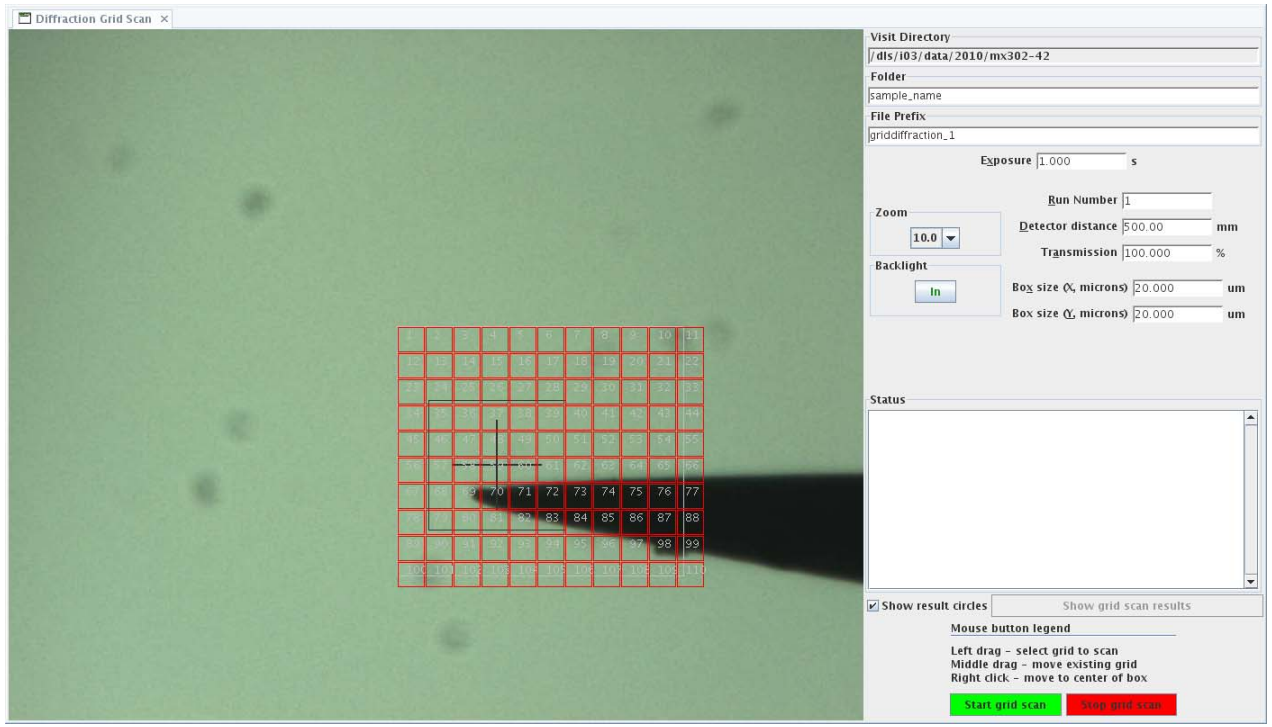
MX End Station



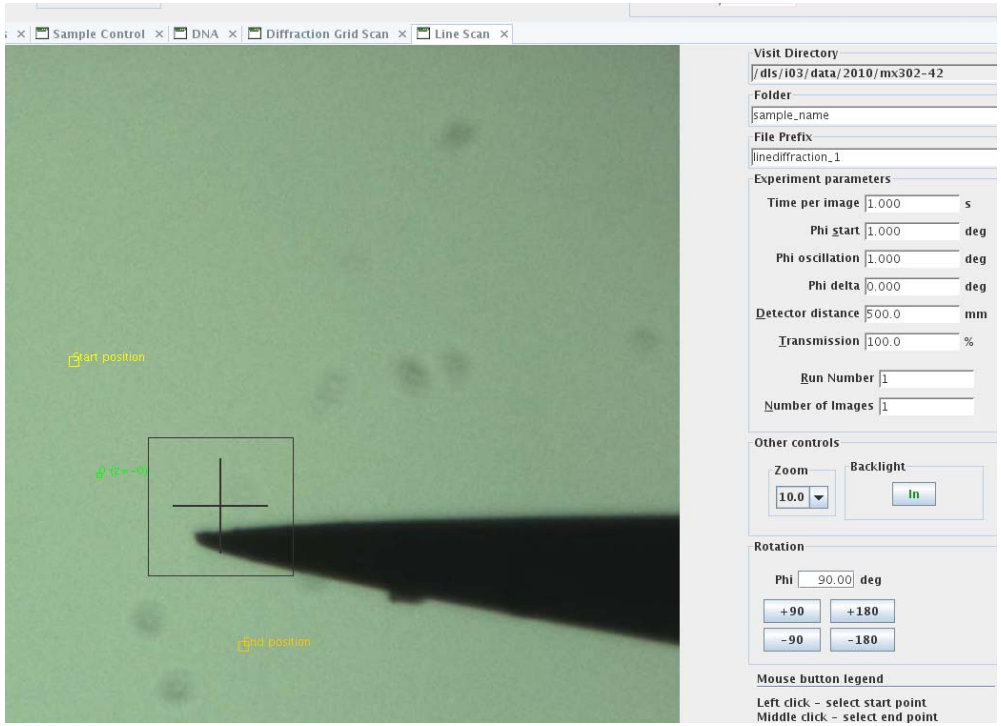
MX Swing GUI


Beamline I03 - Tel. +44 1235 778707 GDA - 8.4.0 - logged in as tcv94921
File View Look and Feel Layout Help
Status
Baton
Health
Experiment Shutter
Closed
Make Hutch Safe
Activate Hutch
Beamline idle
Sample Control idle
Running Script
Synchrotron
Ring Current 150.05 mA
ID Gap 5.30 mm
Time to refill 286 s
Experiment Control
Fluorescence Scans
Sample Control
DNA
Line Scan
Still
Exposure 1.000 s
Take Still
Detector Options
Detector Distance
800.00 mm
Energy 12700.0 eV
Wavelength 0.9763 Angstrom
Auto Alignment on
Re-align Now
Visit Directory
/dls/i03/data/2010/mx302-42
Default Folder
Steve/Mhf12
Default File Prefix
xtal22
Data Collection Parameters
Input
Execution Queue
History
SampleID
Start Data Collection
Pause Data Collection
Stop After Current Image
Panic Stop
Options
Feedback from Data Collection Script
Phi Angle 0.00
Beam Position X 0.00 mm
Holder
Beam Position Y 0.00 mm
Sample
Last Image
Filename
Message
2010-04-15 09:17:52,648 DEBUG Creating i03/epicsTca01 adapter of type gda.device.corba.impl.DeviceAdapter

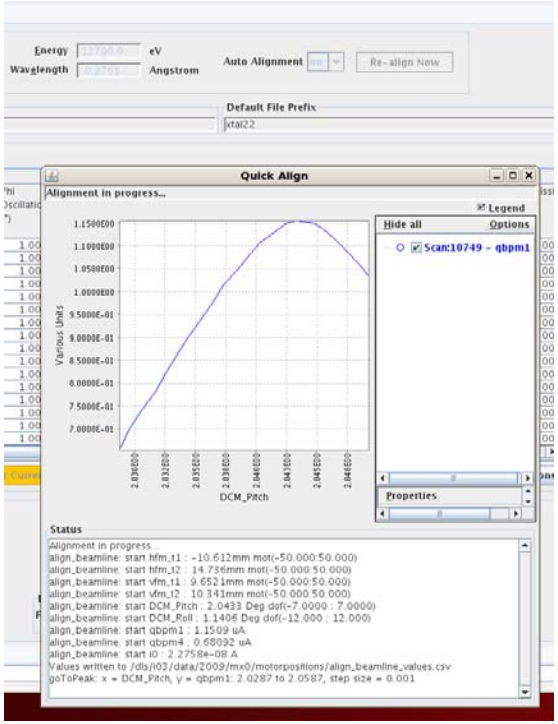
MX Grid Scan


Diffraction Grid Scan
Visit Directory
/dls/i03/data/2010/mx302-42
Folder
sample_name
File Prefix
griddiffraction_1
Exposure 1.000 s
Run Number 1
Zoom
10.0
Detector distance 500.00 mm
Transmission 100.000 %
Backlight
In
Box size (X, microns) 20.000 um
Box size (Y, microns) 20.000 um
Status
Show result circles
Show grid scan results
Mouse button legend
Left drag - select grid to scan
Middle drag - move existing grid
Right click - move to center of box
Start grid scan
Stop grid scan

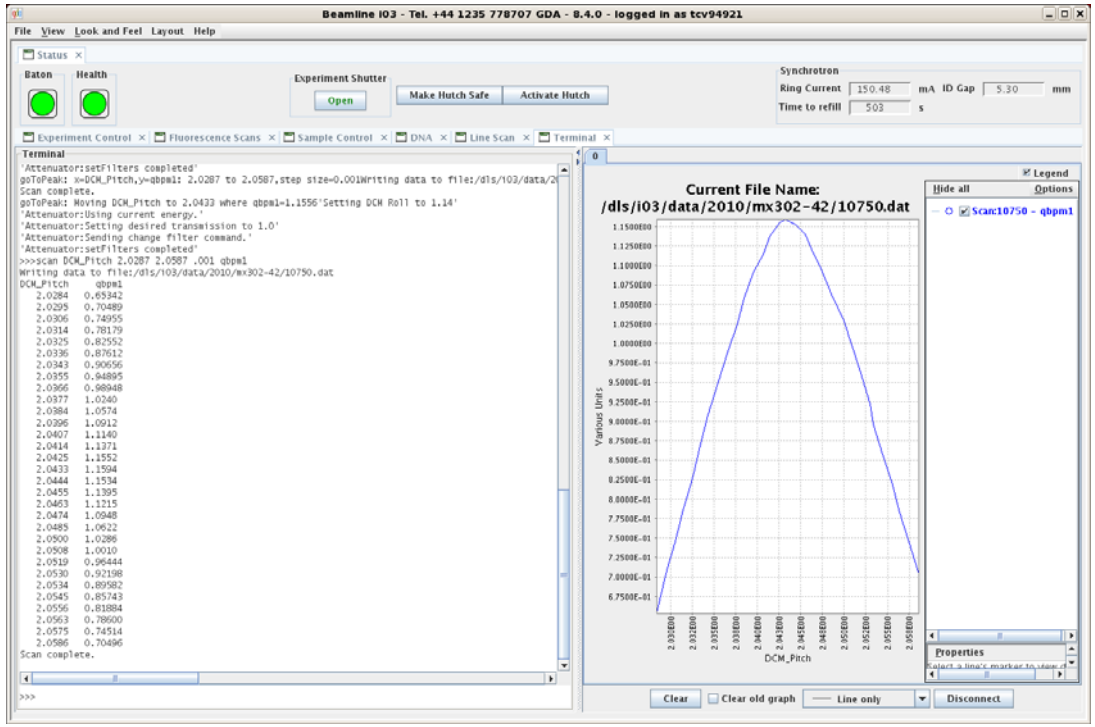
MX Line Scan



MX Automatic Beamline Alignment



MX Simple Scan of DCM Pitch measuring beam current



MX View of experiment history as stored in ISPyB

Visit Directory

/dls/i03/data/2010/mx302-42

Default F

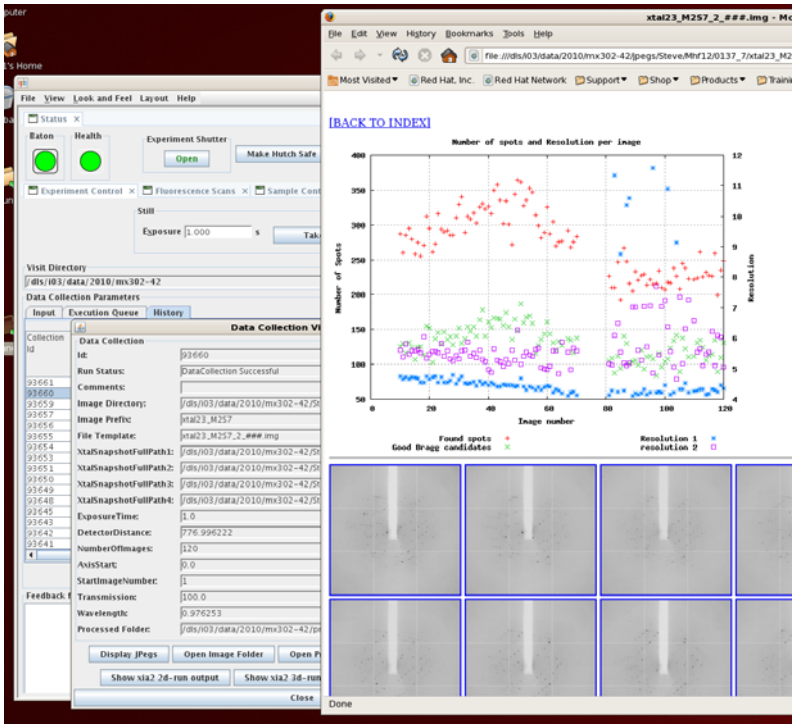
Steve/Mhf

Data Collection Parameters

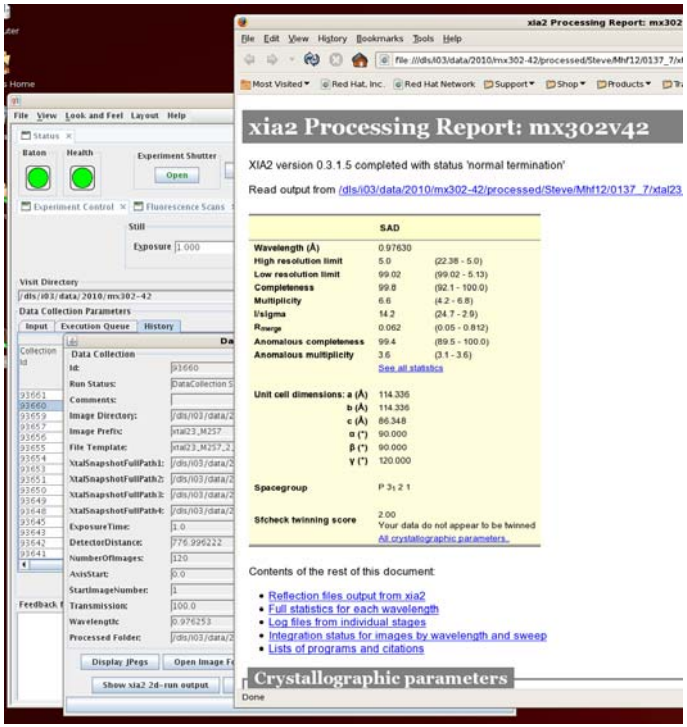
Input | Execution Queue | History

Collection Id	Run Status	Sample ID	Code	Folder	Prefix	Holder	Position
93661	DataColle...			Steve/Mhf...	xtal22_M...		
93660	DataColle...			Steve/Mhf...	xtal23_M...		
93659	DataColle...			Steve/Mhf...	xtal23_M...		
93657	DataColle...			Steve/Mhf...	xtal21_M...		
93656	DataColle...			Steve/Mhf...	xtal21_M...		
93655	DataColle...			Steve/Mhf...	xtal20_M...		
93654	DataColle...			Steve/Mhf...	xtal20_M...		

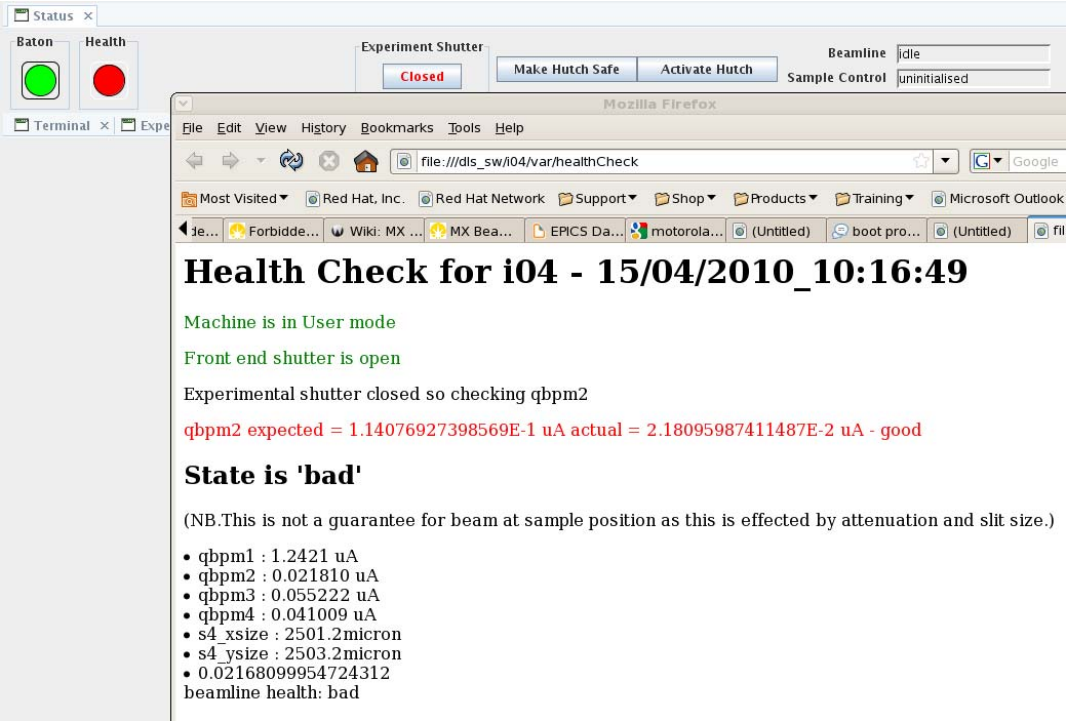
MX View of automatic analysis results (1)



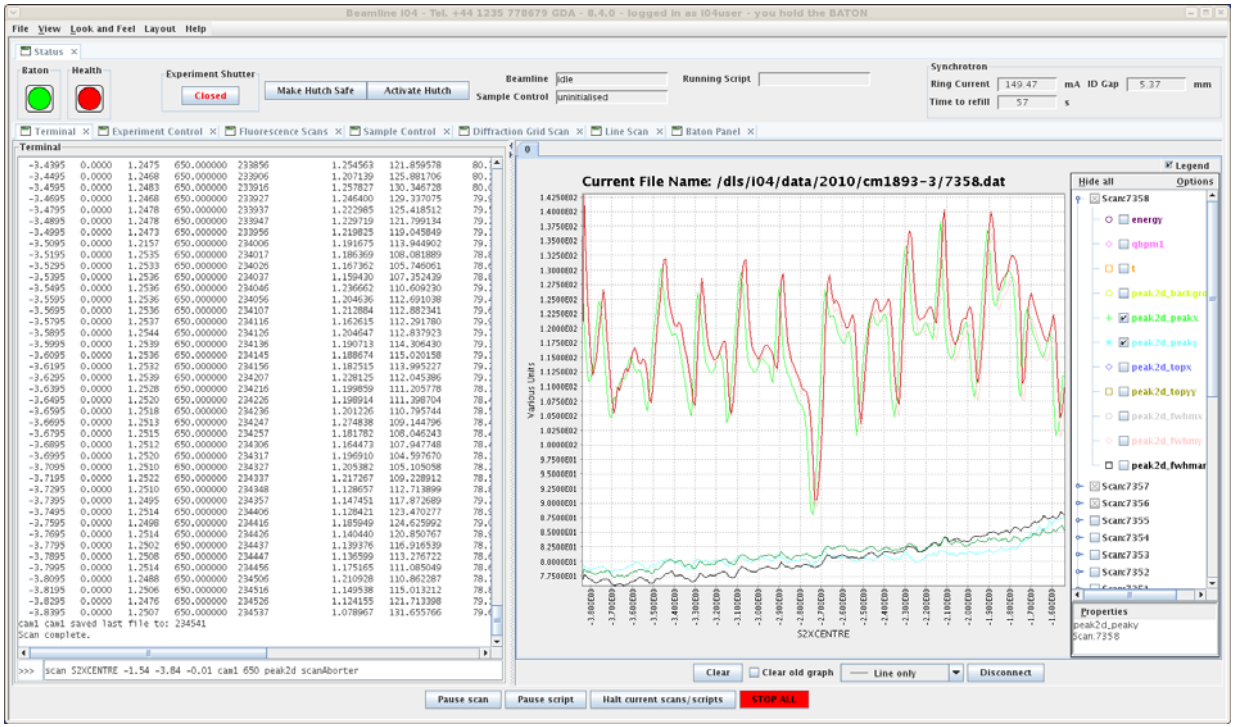
MX View of automatic analysis results (2)



MX - background detection of problems and reporting



Beamline Optimisation - Mirror Focussing



Mirror Focussing - details

Scan Command

```
cam1 cam1 saved last file to: 234541
Scan complete.
>>> scan S2XCENTRE -1.54 -3.84 -0.01 cam1 650 peak2d scanAborter
```

Top level command

```
>>> getData(bm_vfm,-100,S3YGAP,S3YCENTRE,0.37,S2XGAP,S2XCENTRE,-3.89,-5.24,-0.03,863)
```

Mirror Focussing - top level script details (1)

```
def getData(mirror, increment,
                    slitToScanSize,
                    slitToScanPos,
                    slitSize,
                    otherSlitSize,
                    otherSlitPos,
                    slitStart,
                    slitEnd,
                    slitStep,
                    exposure=768):
          positions = getPositions(mirror.getPosPlusIncrement(0),increment)
          print positions
          pos otherSlitPos 0.
          pos otherSlitSize 40.
          pos slitToScanSize slitSize
          for position in positions:
                    print "moving mirror to " + `position`
                    pos mirror position
                    print "mirror is now at " + `mirror()`
                    while(True):
                              while( not scanAborter.isOK()):
                                        print "waiting for beam to come back"
                                        sleep(60)
                              try:
                                        print "beam is ok so now start scan"
                                        scan slitToScanPos slitStart slitEnd slitStep cam1 exposure peak2d scanAborter
                                        break
                              except:
                                        if( scanAborter.isOK()):
                                                  raise Exception, "Scan aborted for unknown reason"
                                        else:
                                                  print "scan aborted due to drop in beam current"
```

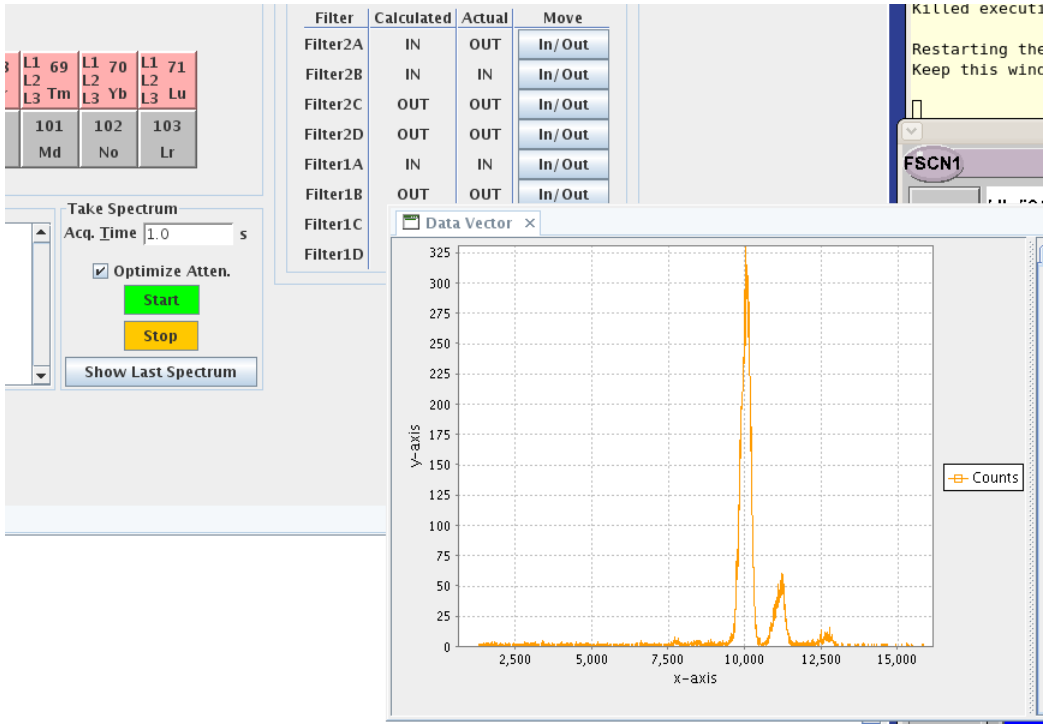
Mirror Focussing - top level script details (2)

```
class ScanAborter(PseudoDevice):
        def __init__(self, name, qbpm, minValue):
                self.qbpm = qbpm1
                self.minValue=minValue
                self.setName(name)
                self.setInputNames(["energy"])
                self.setExtraNames([self.qbpm.getName()])
                self.energy=0.
        def rawIsBusy(self):
                if not self.isOK():
                        raise "qbpm is < " + `self.minValue`
                return False
        def getPosition(self):
                return (self.energy, self.qbpm())
        def isOK(self):
                return self.qbpm() > self.minValue
        def asynchronousMoveTo(self, new_energy):
                self.energy = new_energy
                pass

scanAborter = ScanAborter("scanAborter",qbpm1, 0.5)
```

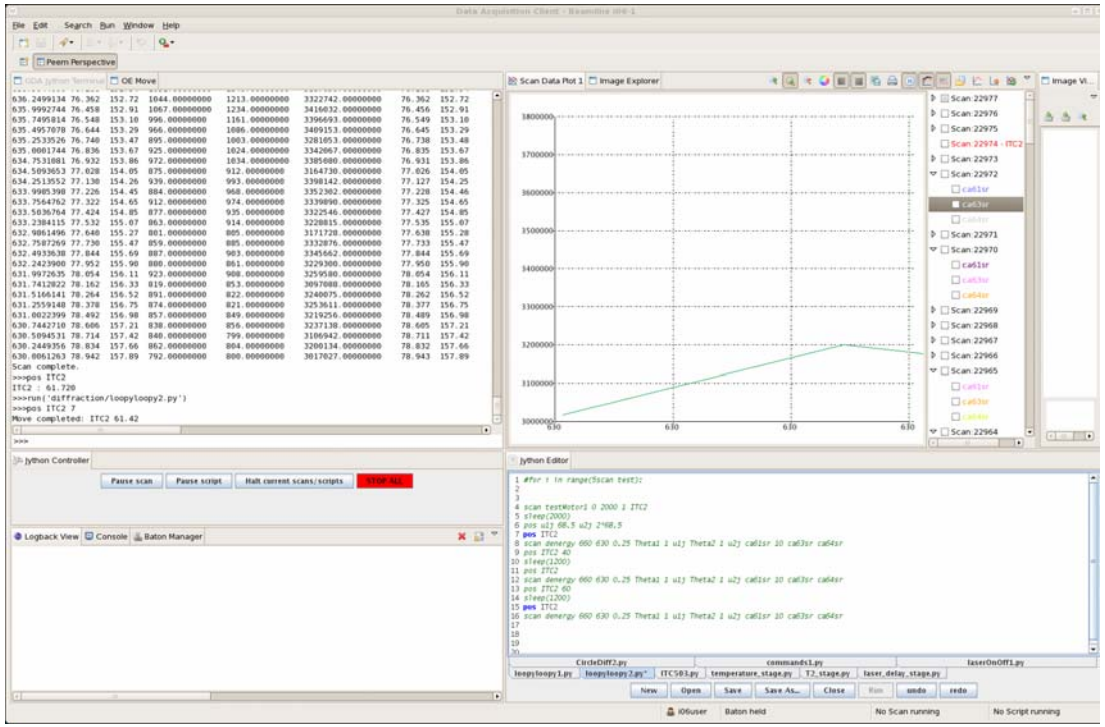
Asking user for input during a script

```
>>>ro=gdascripts.bimorph.bimorph.runOptimisation()
Please enter the type of mirror,either vfm or hfm-->hfm
Please enter the mirror length

-->
```

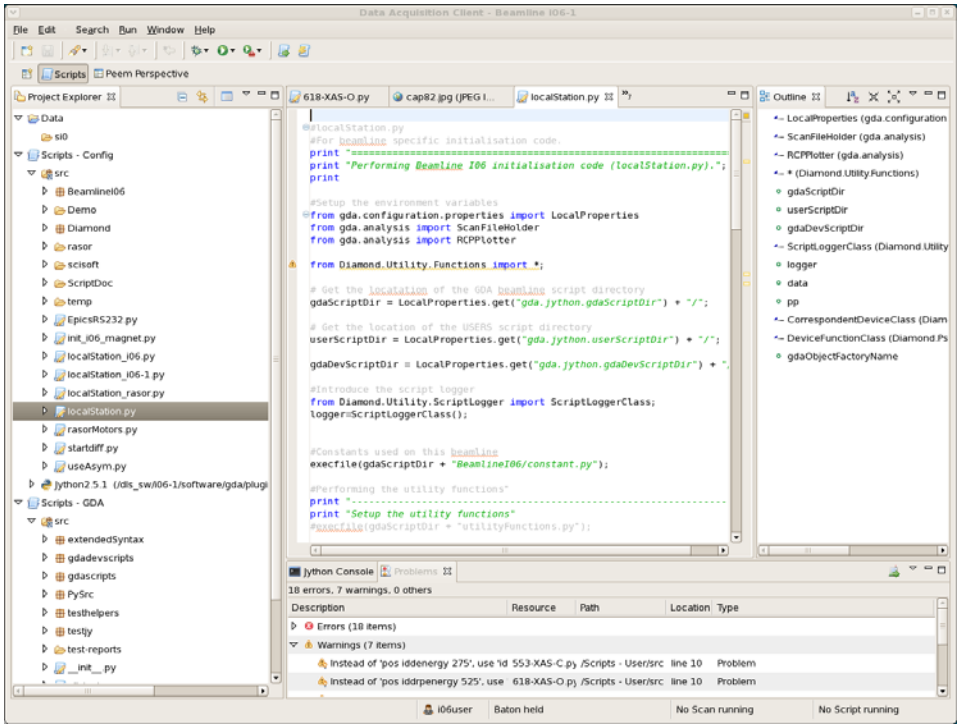
MX - simple visualisaton of scan results



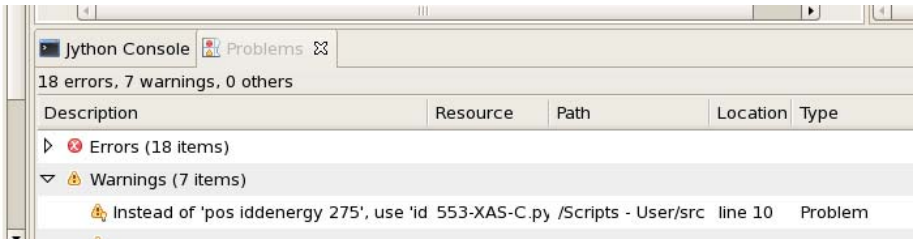
I06 - Eclipse GUI used in script mode



I06 - PyDev editor in action

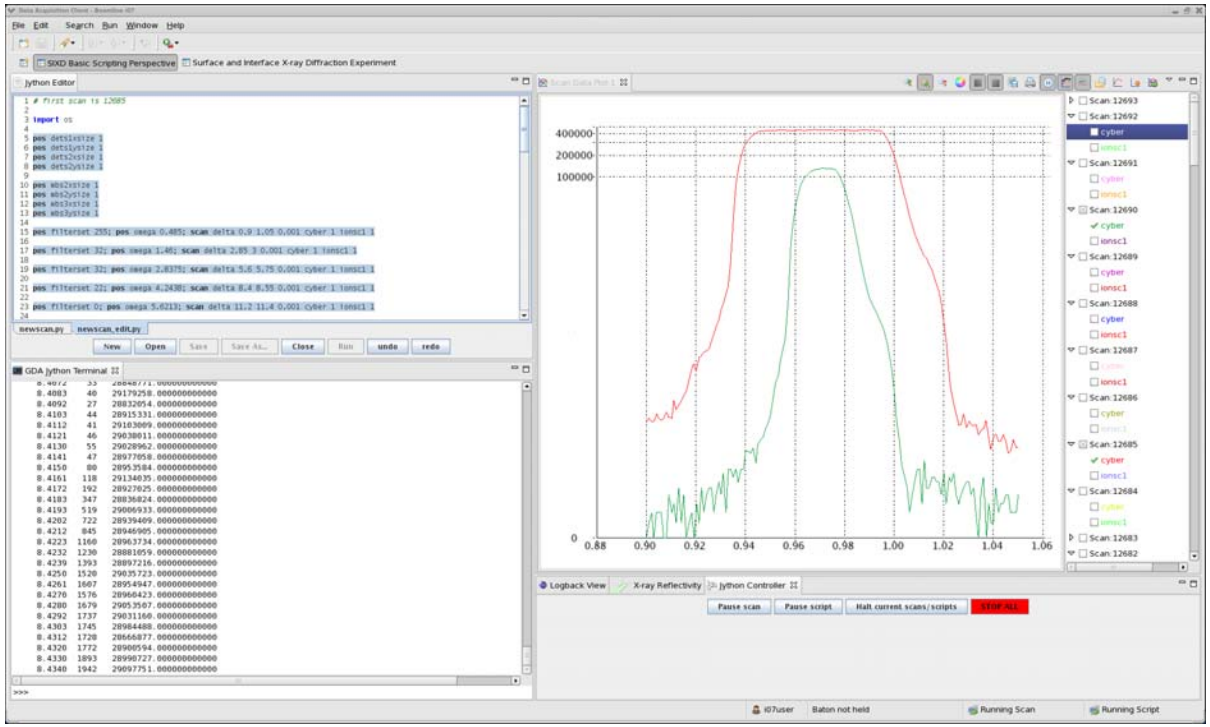


I06 - PyDev editor extended for use in GDA

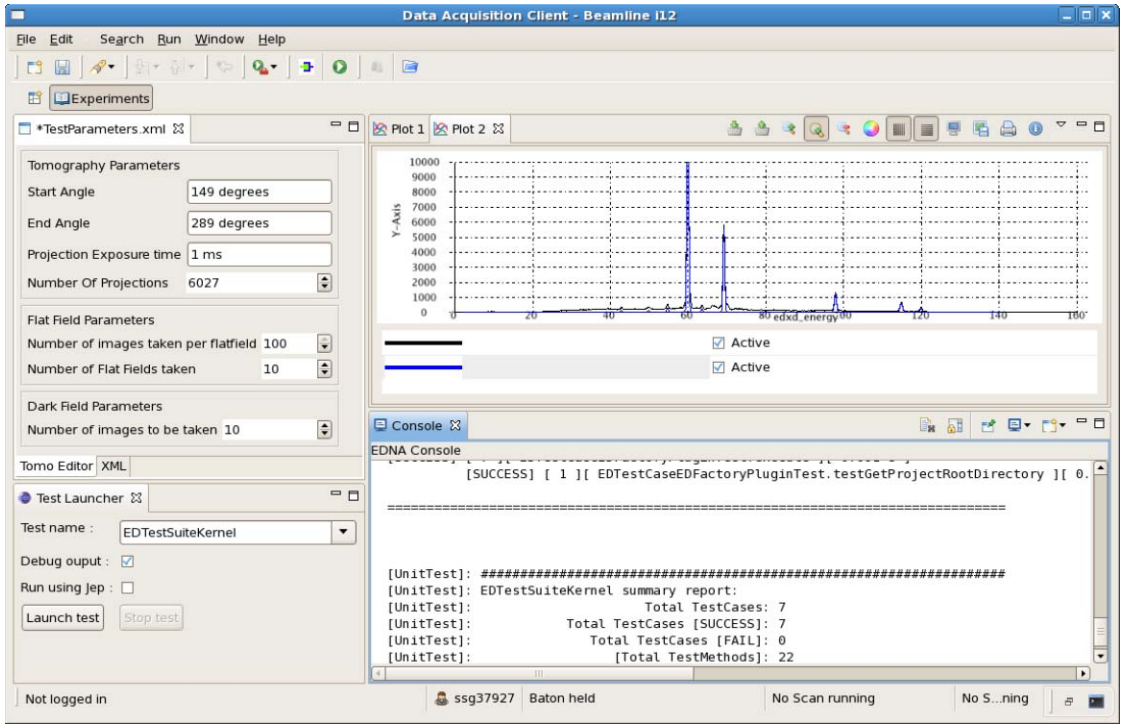


I07 - User Environment

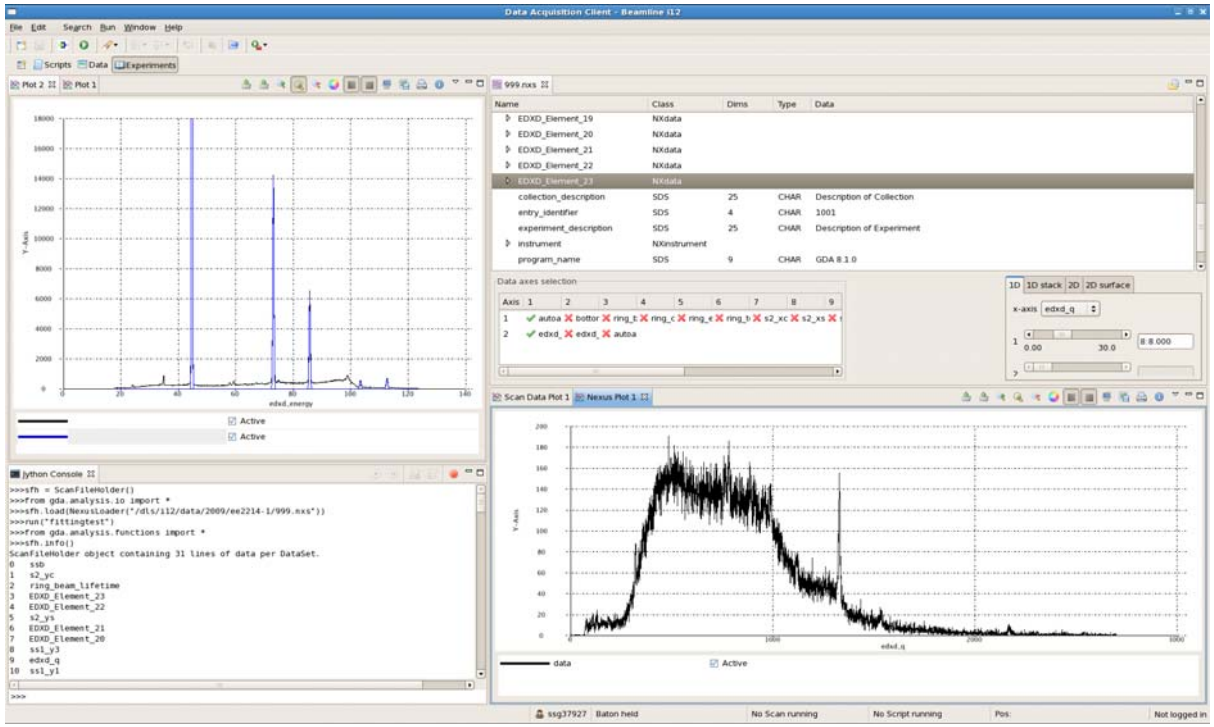
I07 - Scan and Plot



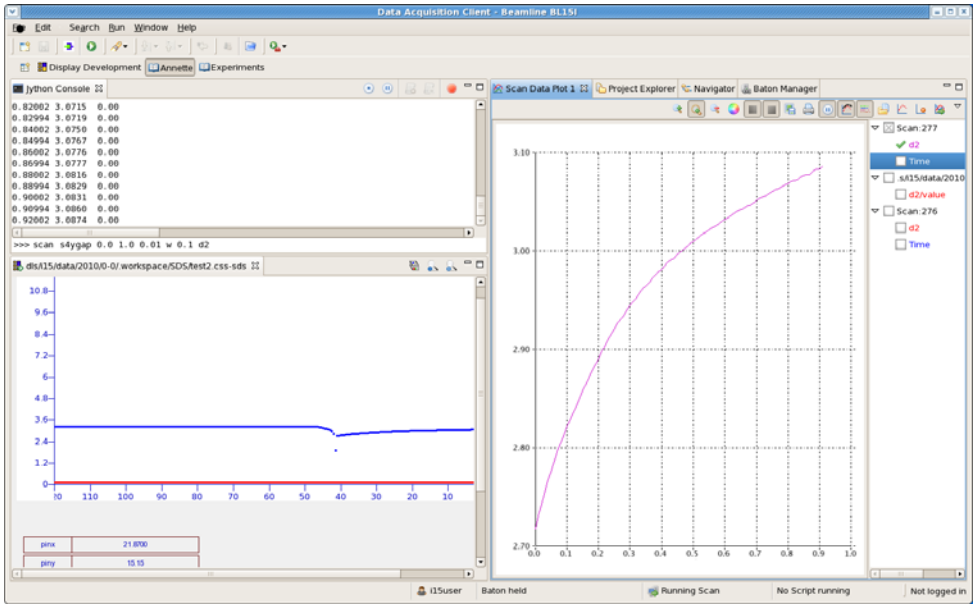
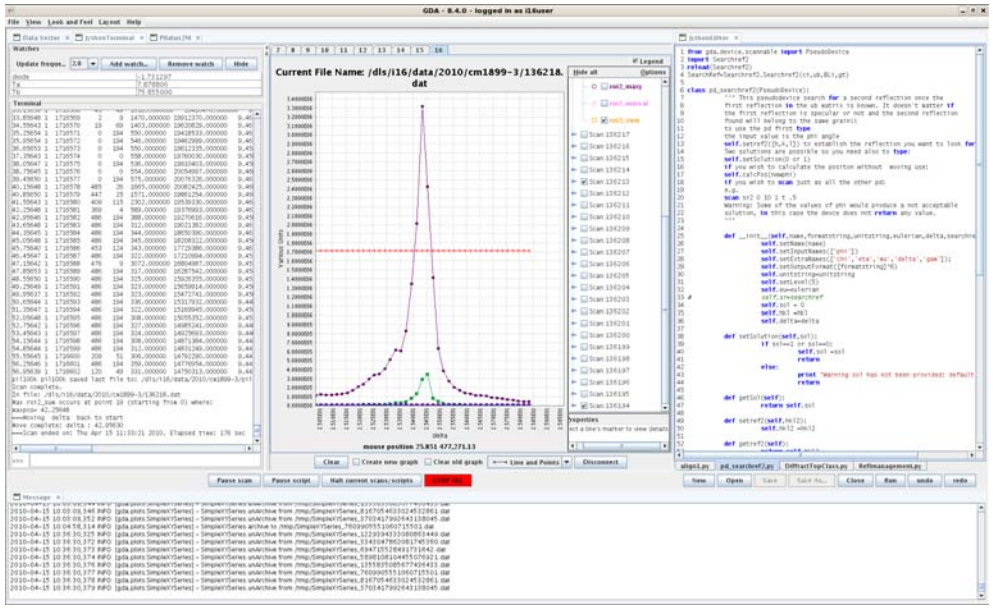
I12 - Experiment Parameter Editor, incorporation of EDNA



I12 - Data analysis, Nexus Viewer and Peak finding



I15 - CSS -SDS panel created by beamline scientist



I16 - Scanning of user written jython scannable